CLAIMS:

1. A method for operation of a dialog system (1) with a speech input interface (2) and an application (3) co-operating with the speech input interface (2) in which the speech input interface (2) detects audio-speech signals (AS) from a user and converts these directly into a recognition result (ER) in the form of binary data which can be used directly by the application (3).

2. A method as claimed in claim 1, characterized in that the binary data comprises at least one program module (PM) present in machine language and executable directly by the application (3) in the form of an object of an object-oriented translator language and/or a data object of an object-oriented translator language.

3. A method as claimed in claim 1 or 2, characterized in that on conversion of the audio speech signal (AS) into a recognition result (ER) first in a syntax analysis step (SA) the phrase corresponding to the audio speech signal (AS) is detected on the basis of a formal grammar (GR) where the valid vocabulary of the audio speech signal (AS) corresponds to the terminal words of the formal grammar (GR), and then the recognition result (ER) is generated in a semantic synthesis step (SS) from the executable program modules (PM) present in machine language and allocated to the terminal words.

4. A method as claimed in claim 3, characterized in that the grammar (GR) is defined completely before the start of a dialog and cannot be changed during the dialog.

5. A method as claimed in claim 3, characterized in that the grammar (GR) is dynamically changed during a dialog.

6. A method as claimed in any one of claims 3 to 5, characterized in that the grammar (GR) comprises substitution rules (AR, KR) which are implemented as object-oriented grammar classes (GK) each of which have a rule-dependent parsing function as a method.

7. A method as claimed in any of claims 3 to 6, characterized in that the grammar (GR) is specified in the form of at least one grammar object (GO) as an instance of at least one object-oriented grammar class (GK) and in the syntax analysis step (SA) the audio speech signal (AS) is checked according to substitution rules (AR, KR) of the grammar (GR).

8. A method as claimed in any of claims 3 to 7, characterized in that the syntax analysis step (SA), the semantic synthesis step (SS) and/or the use/execution of the recognition result (ER) take place at least partly overlapping temporally.

9. A method as claimed in any of claims 6 to 8, characterized in that a program part of the speech input interface generating the recognition result is linked as a method of an object-oriented class, in particular as a method of the grammar object (GO).

10. A method as claimed in any of claims 6 to 8, characterized in that the recognition result (ER) is defined by a method of a grammar class (GR) and returned by this as an object.

11. A method for production of a speech input interface (2) for a dialog system (1) with an application (3) co-operating with the speech input interface (2) and comprising the steps:

- specification of valid speech input signals (AS) by a formal grammar (GR), where the valid vocabulary of the speech input signal is defined in the form of terminal words of the grammar (GR),
- provision of binary data representing the semantics of valid audio speech signals (AS) and comprising data structures which are directly usable by the application (3) at system run time and generated by a program part of the speech input interface (2) and/or program modules (PM) directly executable by the application (3), and/or the provision of program parts which generate the binary data
- allocation of the binary data and/or program parts to individual or combinations of terminal words or non-terminal words to reflect a valid audio speech signal (AS) in appropriate semantics,
- translation of the program parts and/or program modules (PM) into machine language such that on operation of the dialog system (1), the translated program parts

generate data structures directly usable by the application (3) or on operation of the dialog system (1), the translated program modules (PM) can be executed directly by the application (3).

12. A method as claimed in claim 11, characterized in that the formal grammar (GR) is specified by at least one grammar object (GO) as an instance of at least one object-oriented grammar class (GK).

13. A method as claimed in claim 12, characterized in that at least one grammar class (GK) is derived by inheritance from one or more prespecified classes of a grammar class hierarchy and/or a grammar class library.

14. A method as claimed in any of claims 11 to 13, characterized in that the program modules (PM) are programmed in an object-oriented translator language.

15. A method as claimed in any of claims 12 to 14, characterized in that at least one grammar class (GK) and/or the program module (PM) are translated into machine language and provided as static and/or dynamic linked libraries.

16. A method as claimed in any of claims 11 to 15, characterized in that the formal grammar (GR) is specified using a graphic grammar editor and the semantics are defined using a graphic semantic editor.

17. A method as claimed in claim 16, characterized in that the formal grammar (GR) is specified using the graphic grammar editor by selection and/or derivation from prespecified grammar classes (GK) and occupation of the grammar classes with substitution rules (AR, KR) and/or terminal words and/or non-terminal words, where a graphic symbol is allocated to each grammar class (GK) and/or each substitution rule (AR, KR).

18. A method as claimed in claim 16 or 17, characterized in that for definition of the semantics of the formal grammar (GR), for each program module (PM) the graphic semantic editor provides an editor window for production of the program module (PM) and associates the program module with a terminal or non-terminal word.

19. A method to generate a dialog system (1) with a speech input interface (2) and an application (3), where the speech input interface (2) is generated with a method as claimed in any of claims 10 to 17.

20. A method as claimed in claim 19, characterized in that the speech input interface (2), the application (3) and where applicable the program modules (PM) belonging to the recognition result (ER) are each written at least partly in the same object-oriented translator language or can run on the same object-oriented platform.

21. A speech input interface (2) for a dialog system (1) for voice control of a device or method by a user, which with co-operates with an application (3) of the dialog system (1) and detects audio speech signals (AS) and converts these directly into a recognition result (ER) in the form of binary data which can be used directly by the application (3).

22. A dialog system (1) comprising a speech input interface (2) as claimed in claim 21.

23. A system for production of a speech input interface (2) of a dialog system (1) comprising a syntax specification tool with which valid audio signals (AS) of the dialog system (1) are specified by a formal grammar (GR), where the valid vocabulary of the audio speech signal (AS) is defined in the form of terminal words of the grammar (GR), and a semantic definition tool for provision of program modules (PM) and allocation of the program modules (PM) to individual or combinations of terminal words such that after translation into machine language, on operation of the dialog system (1) the translated program modules (PM) can be executed directly by the application (3).

24. A system as claimed in claim 23, characterized by an object-oriented grammar class library and/or an object-oriented grammar class hierarchy so that the formal grammar (GR) is specified as an instance of a grammar class (GK) taken from a grammar class library or derived from the classes of the grammar class library, and/or as an instance of a grammar class (GK) taken from the grammar class hierarchy or derived from the classes of the grammar class hierarchy.

25. A system as claimed in claim 23 or 24, characterized by a graphic grammar editor to specify the formal grammar (GR) and/or a graphic semantic editor to define the semantics.